```
#include <stdio.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <string.h>
// structure definition for accepting the packets.
struct frame
{
    int packet[40];
};
// structure definition for constructing the acknowledgement frame
struct ack
{
    int acknowledge[40];
};
int main()
{
    int clientsocket;
    struct sockaddr_in serveraddr;
    socklen_t len;
    struct hostent *server;
    struct frame f1;
    int windowsize, totalpackets, totalframes, i = 0, j = 0, framesreceived = 0,
k, l, buffer;
    struct ack acknowledgement;
    char req[50];
    clientsocket = socket(AF_INET, SOCK_DGRAM, 0);
    bzero((char *)&serveraddr, sizeof(serveraddr));
    serveraddr.sin_family = AF_INET;
    serveraddr.sin_port = htons(5018);
    server = gethostbyname("127.0.0.1");
    bcopy((char *)server->h_addr, (char *)&serveraddr.sin_addr.s_addr,
          sizeof(server->h_addr));
    // establishing the connection.
    printf("sending request to the server\n");
    sendto(clientsocket, "HI IAM CLIENT", sizeof("HI IAM CLIENT"), 0,
           (struct sockaddr *)&serveraddr, sizeof(serveraddr));
    printf("\nWaiting for reply\n");
    recvfrom(clientsocket, req, sizeof(req), 0,
             (struct sockaddr *)&serveraddr, &len);
    printf("\n The server has to send :\t%s\n", req);
    // accepting window size from the user.
    printf("\nenter the window size\n");
    scanf("%d", &windowsize);
    // sending the window size.
    printf("\n sending window size\n");
    sendto(clientsocket, (char *)&windowsize, sizeof(windowsize), 0,
           (struct sockaddr *)&serveraddr, sizeof(serveraddr));
    // collecting details from server.
    printf("\n waiting for the server response\n");
    recvfrom(clientsocket, (char *)&totalpackets, sizeof(totalpackets), 0,
             (struct sockaddr *)&serveraddr, &len);
    printf("\nTotal packets are :\t%d\n", totalpackets);
    sendto(clientsocket, "RECEIVED", sizeof("RECEIVED"), 0,
           (struct sockaddr *)&serveraddr, sizeof(serveraddr));
    recvfrom(clientsocket, (char *)&totalframes, sizeof(totalframes), 0,
             (struct sockaddr *)&serveraddr, &len);
    printf("\n total number of frames or windows are:\t%d\n", totalframes);
    sendto(clientsocket, "RECEIVED", sizeof("RECEIVED"), 0,
           (struct sockaddr *)&serveraddr, sizeof(serveraddr));
    // starting the process.
    printf("\nstarting the process of receiving\n");
    while (i < totalpackets)
    {
```

```
        // initialising the receive buffer.
        printf("\nInitializing the received buffer\n");
        printf("\nThe expected frame is %d with packets:", framesreceived);
        j = 0;
        buffer = i;
        while (j < windowsize && i < totalpackets)
        {
            printf("%d", i);
            i++;
            j++;
        }
        printf("\nwaiting for the frame\n");
        // accepting the frame.
        recvfrom(clientsocket, (char *)&f1, sizeof(f1), 0,
                 (struct sockaddr *)&serveraddr, &len);
        printf("\n received frame %d\n\n enter -1 to send negative
acknowledgement for the following packets \n", framesreceived);
        // constructing the acknowledgement frame.
        j = 0;
        l = buffer;
        k = 0;
        while (j < windowsize && l < totalpackets)
        {
            printf("\npacket:%d\n", f1.packet[j]);
            // accepting acknowledgement from the user.
            scanf("%d", &acknowledgement.acknowledge[j]);
            if (acknowledgement.acknowledge[j] == -1)
            {
                if (k == 0)
                {
                    i = f1.packet[j];
                    k = 1;
                }
            }
            j++;
            l++;
        }
        framesreceived++;
        // sending acknowledgement to the server.
        sendto(clientsocket, (char *)&acknowledgement, sizeof(acknowledgement),
0,
               (struct sockaddr *)&serveraddr, sizeof(serveraddr));
    }
    printf("\nall frames received successfully\n closing connection with the
server\n");
    close(clientsocket);
}
```